Artificial Intelligence and Robotics Research 人工智能与机器人研究, 2025, 14(2), 362-375
Published Online March 2025 in Hans. https://www.hanspub.org/journal/airr
https://doi.org/10.12677/airr.2025.142036

Hans汉斯

基于两阶段蒸馏的动作识别

陈　凯，党存远，蔡子当，夏雨涵，孙永宣

合肥工业大学计算机与信息学院，安徽　合肥

收稿日期：2025年2月17日；录用日期：2025年3月14日；发布日期：2025年3月25日

摘　要

在计算机视觉领域，CNN与Transformer分别在局部信息提取和全局特征建模方面具有优势，如何融合CNN与Transformer成为研究热点之一。一些工作直接在Transformer编码器中引入卷积运算，然而这会改变Transformer的原有结构，限制自注意力的全局建模能力。另一些工作在CNN与Transformer的log*it*输出层进行知识蒸馏，然而其未能利用CNN的特征层信息。针对上述问题，本文提出特征对齐蒸馏模块，通过将Transformer的特征层与CNN的特征层进行维度对齐，实现了Transformer与CNN的特征层蒸馏，使Transformer学习到了CNN的局部建模能力。针对特征对齐操作会引入卷积操作增加模型计算量的问题，本文又提出了特征映射log*it*蒸馏模块，通过将Transformer的特征层映射为log*it*，实现了Transformer与CNN特征层的通用蒸馏方法。为了使学生模型同时学习局部建模能力和长距离依赖建模能力，本文提出了两阶段蒸馏框架，实现了CNN教师和Transformer教师对学生模型的协同指导。实验结果表明，本文方法实现了CNN与Transformer的特征层蒸馏，并使学生模型在CNN教师和Transformer教师的协同指导下，同时学习到了局部建模能力和长距离依赖建模能力，提高了基准模型在动作识别下游任务上的准确率。

关键词

特征蒸馏，模型融合，两阶段蒸馏，动作识别

Action Recognition Based on Two-Stage Distillation

Kai Chen, Cunyuan Dang, Zidang Cai, Yuhan Xia, Yongxuan Sun

School of Computer Science and Information Engineering, Hefei University of Technology, Heifei Anhui

Received: Feb. 17th, 2025; accepted: Mar. 14th, 2025; published: Mar. 25th, 2025

Abstract

In the field of computer vision, CNN and Transformer have advantages in local information extraction and global feature modeling, respectively, and how to fuse CNN and Transformer has become one of

文章引用: 陈凯, 党存远, 蔡子当, 夏雨涵, 孙永宣. 基于两阶段蒸馏的动作识别[J]. 人工智能与机器人研究, 2025, 14(2): 362-375. DOI: 10.12677/airr.2025.142036

the research hotspots. Some works directly introduce convolutional operations in the Transformer encoder, however, this will change the original structure of the Transformer and limit the global modeling ability of self-attention. Some other work performs knowledge distillation in the log*it* **output layer of CNN and Transformer, however, it fails to utilize the feature layer information of CNN. Aiming at the above problems, this paper proposes the feature alignment distillation module, which realizes the feature layer distillation between Transformer and CNN by dimensionally aligning Transformer's feature layer with CNN's feature layer, so that Transformer learns the CNN's local modeling ability. Aiming at the problem that the feature alignment operation will introduce the convolution operation to increase the model computation, this paper also proposes the feature mapping log*it*** **distillation module, which realizes a general distillation method for the feature layer of Transformer and CNN by mapping the feature layer of Transformer to log*it*. In order to enable student models to learn both local modeling ability and long-distance dependent modeling ability, this paper proposes a two-stage distillation framework, which realizes the collaborative guidance of CNN teachers and Transformer teachers to student models. The experimental results show that the method in this paper achieves feature layer distillation of CNN and Transformer, and enables the student model to learn both local modeling capability and long-distance dependency modeling capability under the collaborative guidance of CNN instructor and Transformer instructor, which improves the accuracy of the baseline model on the downstream task of action recognition.**

Keywords

Feature Distillation, Model Fusion, Two-Stage Distillation, Action Recognition

1. 引言

在计算机视觉领域，卷积神经网络(CNN)和 Transformer 网络是两种常见模型，他们具有各自的优势。CNN 由于卷积运算的局部性和权值共享机制，具有很强的归纳偏置(归纳偏置是指：在不依赖数据的情况下，将学习算法推向特解，从而高度影响学习算法的泛化能力[1] [2])，能够有效地提取图像中的局部特征。而 Transformer 的核心在于自注意力机制，它允许模型在处理序列数据时，能够同时考虑输入序列中所有位置的信息，从而捕捉到长距离依赖关系[3]。因此，为了同时利用 CNN 与 Transformer 的优势，一些工作尝试将 CNN 与 Transformer 模型进行融合[4]。使 Transformer 模仿 CNN 的多尺度金字塔结构，利用 patch embedding 层和加入了空间缩减机制的编码器层实现对特征图尺度的灵活调整。但是，这带来了计算成本高、参数量大等问题[5]。使用先 CNN 再 Transformer 的串联连接方式，将 CNN 网络提取出的低分辨率的特征图重塑为一系列特征序列并对其进行位置编码，将结果输入到 Transformer 中进行学习[6]；则使用先 Transformer 再 CNN 的串联连接方式，其在 vit [7]后串行拼接了 Faster R-CNN [8]作为模型的目标检测网络，并将 vit 的输出重组并输入目标检测网络的残差块。然而，上述两种方式会改变 Transformer 的原有结构，降低 Transformer 的容量(容量是指模型的拟合能力，与模型的参数量和深度有关)。其他方法如 Deit [9]则采用知识蒸馏的方式，通过在输入序列中引入蒸馏 token，将 CNN 教师模型的泛化能力蒸馏到 Transformer 学生模型中。然而，一些研究显示，在 Transformer 网络的早期阶段引入卷积操作能够提高模型的效果[5] [10]，而 Deit [9]仅在教师模型和学生模型的 log*it* 输出层进行蒸馏计算，使得浅层 Transformer 难以学习 CNN 网络的局部建模能力。因此，本文设计了一个 CNN 网络与 Transformer 网络的特征对齐蒸馏模块，在不改变 Transformer 原有结构的同时，使 Transformer 学生模型能够从 CNN 教师模型的特征层学习局部建模能力。

使用特征对齐蒸馏模块，实现了在不改变原有模型网络结构的情况下，在 CNN 与 Transformer 的特征层面进行知识蒸馏，将 CNN 的局部建模能力转移到 Transformer 中。但是，上述模块还存在一些问题。首先，为了对齐 CNN 与 Transformer 的特征层维度，本文构造了如图 1(b)所示的特征对齐模块，而在该模块中引入了卷积运算等操作，增加了模型的计算量。其次，[11]通过 CKA [12] [13] (特征相似度量)方法生成了异构网络的特征相似性热图，并发现来自异构网络的特征存在于不同的潜在特征空间中，因而不能保证成功对齐异构网络的特征。尽管上述特征对齐模块在本文的实验设置中有效地实现了异构网络(即 CNN 与 Transformer)间的特征对齐知识蒸馏，但却缺乏一定的通用性。而相比于网络模型的特征层，log*it* 层直接描述网络的输出分布，具有更强的通用性。因此，为了降低特征对齐的计算量，并实现 CNN 与 Transformer 特征层的通用蒸馏方法，本文提出了特征映射 log*it* 蒸馏模块：通过在 Transformer 学生模型的特征层加入额外的 log*it* 输出分支，实现 Transformer 学生模型的特征层与 CNN 教师模型的 log*it* 输出层之间的蒸馏。

在[14]中，经过预训练的 VideoMAE 教师[15]和 MAE 教师[16]同时对学生模型进行指导，使学生模型同时学习到了空间建模能力和时间建模能力。为了使 Transformer 学生模型同时学习到 CNN 教师的局部建模能力和 Transformer 教师的长距离依赖建模能力，本文提出了一个两阶段蒸馏框架：第一阶段，在 Transformer 学生和 CNN 教师的特征层面进行蒸馏，从而学习 CNN 教师的局部建模能力；第二阶段，引入一个层数更多的 Transformer 教师来指导学生模型，从而学习 Transformer 教师的长距离依赖建模能力。与[14]中学生模型同时与两个教师进行知识蒸馏不同，本文将知识蒸馏分为两个阶段，每个阶段只存在一个教师模型。

本文的主要贡献总结如下：

• 本文提出了特征对齐蒸馏模块，实现了 CNN 与 Transformer 间的特征层对齐，并将 CNN 的局部建模能力转移到了 Transformer 中。

• 本文提出了特征映射 log*it* 蒸馏模块，实现了 CNN 与 Transformer 特征层的通用蒸馏方法。

• 本文提出了两阶段蒸馏框架，通过 CNN 教师与 Transformer 教师的协同指导，使学生模型同时学习到了局部建模能力和长距离依赖建模能力。

2. 方法

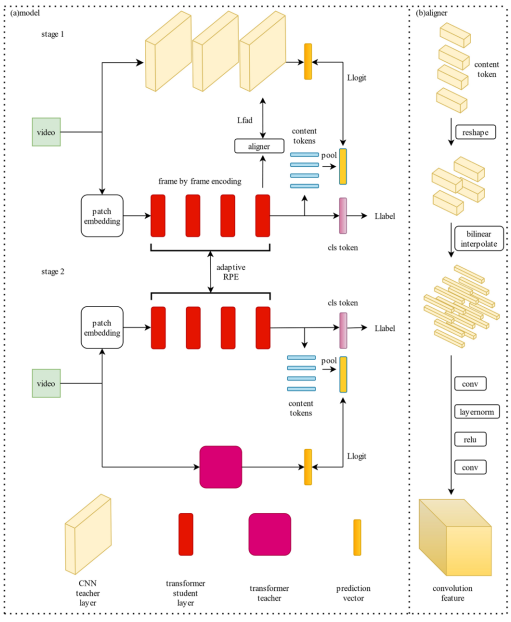

Figure 1. Feature alignment distillation module and overall architecture
图 1. 特征对齐蒸馏模块和整体架构

2.1. 特征对齐蒸馏模块

在之前的工作中，[17]通过研究卷积与自注意力层之间的关系证明了自注意力的卷积等效性：通过设置二次编码，一个具有 N 个头且相对位置编码的维度不小于 3 的多头自注意力层可以表示任何核大小为 $\sqrt{N}\times\sqrt{N}$ 的卷积层。而在[17]中，二次编码即为相对位置编码：

$$v^{(h)} := -\alpha^{(h)}\left(1,-2\Delta_1^{(h)},-2\Delta_2^{(h)}\right)\ r_\delta := \left(\|\delta\|^2,\delta_1,\delta_2\right)\ W_{qry}=W_{key}:=0\ \widehat{W_{key}}:=I \tag{1}$$

在公式(1)中，可学习参数 $\Delta^h=\left(\Delta_1^{(h)},\Delta_2^{(h)}\right)$ 和 $\alpha^{(h)}$ 分别确定了每个头的注意力中心和宽度，而固定参数 $\delta=(\delta_1,\delta_2)$ 则表示 query 和 key 像素之间的相对位移。为了实现 CNN 与 Transformer 特征层的对齐，在[17]的基础上，本文提出了多头卷积注意力层，称为：*MHCA*。原始的多头自注意力计算如公式(2)所示：

$$MHSA(X)=soft\max\left(QK^T\right)V \tag{2}$$

而本文提出的多头卷积注意力层，通过使用相对位置自注意[18]，使得自注意力层表示为卷积层。相比 *MHSA*，*MHCA* 仍然进行线性投影、点积注意力缩放和归一化等操作。不同的是，在得到注意力权重矩阵之前，需要增加参数 $v^{(h)}$ 和 r_δ，以引入相对位置信息。具体来说，对于输入 $X\in R^{n\times d}$，*MHCA* 执行如下多头自注意力计算：

$$MHCA(X)=soft\max\left(QK^T+v^{(h)}r_\delta\right)V \tag{3}$$

在公式(3)，$v^{(h)}$ 和 r_δ 均参照公式(1)。由于 $v^{(h)}$ 包含可学习参数 Δ^h 和 $\alpha^{(h)}$，而 Δ^h 和 $\alpha^{(h)}$ 分别确定了每个头的注意力中心和宽度，所以 *MHCA* 可以自适应学习相对位置嵌入的合适范围[19] (即图 1 中的 adaptive RPE)。在图 1 中，令 Transformer 学生特征层的 content token 大小为 $A^s\in R^{l\times c}$ (l 和 c 分别为 token 的数量和维度)，CNN 教师特征层的 feature map 大小为 $A^t\in R^{h\times w\times c}$ (h、w 和 c 分别为 feature map 的高度、宽度和通道数)。为了对齐 A^s 和 A^t，本文提出了特征对齐模块(即图 1 中(a)所示的 aligner 模块)，其详细结构如图 1(b)所示。对于 Transformer 学生的 content token，特征对齐模块对其进行堆叠重塑、线性插值、卷积、层归一化、relu 函数等操作，以匹配 content token 和 feature map 的大小。在学生模型和教师模型的特征层维度对齐之后，本文使用蒸馏模块将 CNN 教师的局部建模能力蒸馏到 Transformer 学生中，如图 1 中的 L_{fad} 所示：

$$L_{fad}=MSE\left(A^t,aligner\left(A^s\right)\right) \tag{4}$$

在公式(4)中，A^t 和 A^s 分别为 CNN 教师和 Transformer 学生的特征层输出，*aligner* 为特征对齐模块，*MSE* 表示均方差损失。

2.2. 特征映射 log*it* 蒸馏模块

基于特征的蒸馏和基于 log*it* 的蒸馏是两种常见的蒸馏方法，它们各有优势。基于特征的蒸馏的目标是使得学生模型的中间层特征尽可能接近教师模型的中间层特征，以帮助学生模型学习教师模型各层级的深层次特征。而基于 log*it* 的蒸馏的目标则是尽可能地匹配学生模型与教师模型的输出分布。相比基于特征的蒸馏，其减少了计算的复杂度。在之前的工作中，若进行蒸馏的教师模型和学生模型架构相似(如教师模型与学生模型都是 CNN 架构或都是 Transformer 架构)，它们的学习表征在特征空间中表现出相似性[11]。此时，对教师模型和学生模型的特征层输出使用简单的相似性度量函数(如 *MSE* 均方差函数)就可以取得良好的蒸馏效果[20]。然而，若进行蒸馏的教师模型和学生模型架构不同，由于异构模型的特征存在于不同的特征空间中，因此不能保证成功对齐教师模型与学生模型的特征[11]。尽管在 2.1 节提出的特征对齐蒸馏模块是有效的，但为了提高异构网络(即 CNN 与 Transformer)特征层蒸馏的通用性，同时为

了避免特征对齐模块引入的额外卷积运算，本文提出了 CNN 与 Transformer 的特征映射 log*it* 蒸馏模块。

与 2.1 节中的特征对齐蒸馏模块不同，特征映射 log*it* 蒸馏模块不是直接对教师模型和学生模型的特征层输出进行蒸馏，而是通过在学生模型的特征层加入额外的 log*it* 输出分支，将学生模型特征层与教师模型特征层不匹配的表示转移到对齐的 log*it* 空间中。通过对学生模型额外的 log*it* 输出分支与教师模型的 log*it* 输出进行损失计算，使学生模型的特征层学习教师模型的预测分布，实现了特征层的通用异构网络蒸馏。

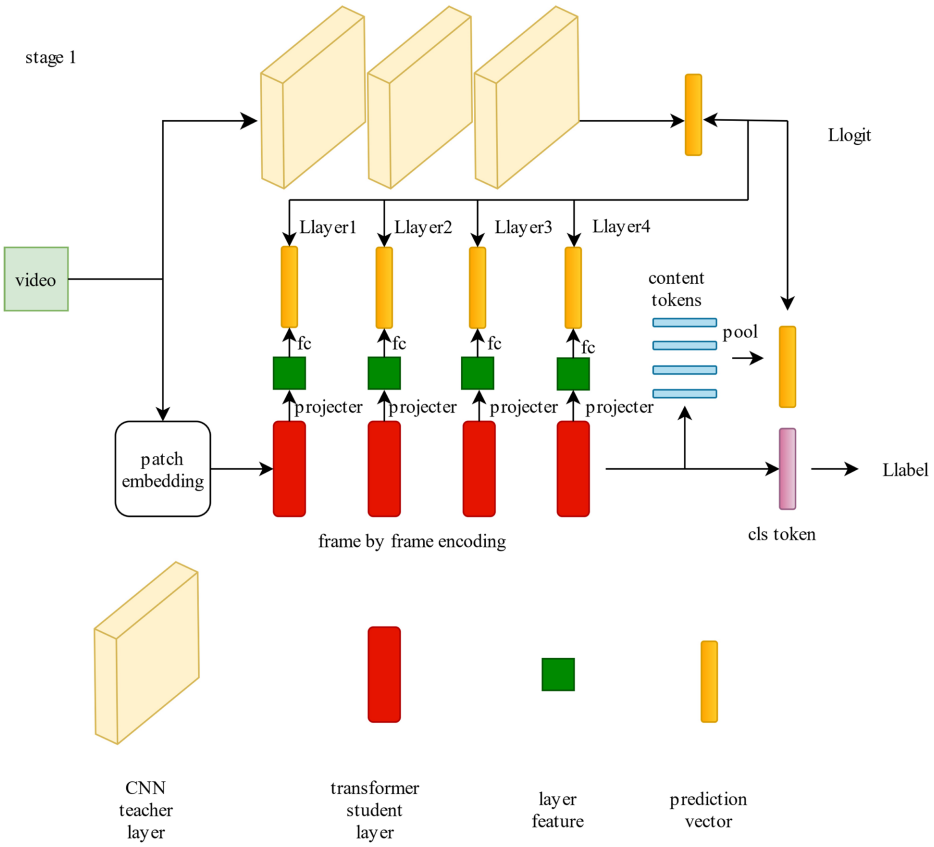

Figure 2. Feature project log*it* distillation module
图 2. 特征映射 log*it* 蒸馏模块

本文提出的特征映射 log*it* 蒸馏模块如图 2 所示。为方便，将 Transformer 学生模型的层数设置为 4。每个特征层经过投影、全连接操作后得到一个预测向量，此向量即为特征层的 log*it* 得分。对于 CNN 教师模型，一方面，CNN 教师模型的层数与 Transformer 学生模型的层数不一致，在与学生模型的特征层蒸馏时难以确定蒸馏点(即蒸馏层数的匹配)；另一方面，由于特征映射 log*it* 蒸馏模块最终是使用教师模型的 log*it* 得分与学生模型进行蒸馏，而教师模型最后一层的 log*it* 得分效果必然优于特征层的 log*it* 得分效果，因此，本文直接使用教师模型最后一层 log*it* 得分与学生模型的特征层 log*it* 得分进行蒸馏。

尽管不同架构的模型在 log*it* 空间中学习相同的目标，但这些模型拥有不同的归纳偏置[19]，这使得它们得到不同的预测分布。为了缓解这个问题，仿照[11]，本文通过将目标类和非目标类的 log*its* 信息分开引导，用以增强目标类的信息。原始的 log*it* 蒸馏损失[21]如下所示：

$$L_{KD} = \lambda E_{(x,y)\sim(X,Y)}\left[D_{CE}\left(p^s, y\right) + \left(1-\lambda\right) D_{KL}\left(p^s, p^t\right)\right] \tag{5}$$

在公式(5)中，(X,Y) 是样本和类标签的联合分布，x 是样本输入，y 为真值标签，p^s 和 p^t 分别为学生模型和教师模型对样本输入的预测，D_{CE} 和 D_{KL} 分别为交叉熵损失函数和 KL 散度，λ 则是控制硬标签和软标签蒸馏的系数。通过将目标类和非目标类的 logits 信息分开引导，蒸馏损失可以表示如下：

$$L_{KD1} = -\left(1+p_{\hat{c}}^{t}\right)\log p_{\hat{c}}^{s} - E_{(c\sim Y)/\{\hat{c}\}} p_c^t \log p_c^s \tag{6}$$

在公式(6)中，c 和 $\hat{c}$ 分别表示预测类和目标类，由于 KL 散度的分母项对梯度没有贡献，因此相比于原始蒸馏损失将分母项省去，且为了得到通用的蒸馏损失，将超参数 λ 也省去。最后，为了增强来自目标类的信息，在 $-\left(1+p_{\hat{c}}^{t}\right)$ 项上增加一个调制参数 η，得到如下蒸馏损失：

$$L_{KD2} = -\left(1+p_{\hat{c}}^{t}\right)^{\eta}\log p_{\hat{c}}^{s} - E_{(c\sim Y)/\{\hat{c}\}} p_c^t \log p_c^s \tag{7}$$

在 3.2 节和 3.3 节的实验中，均取 η 为 1.1。本文将 Transformer 学生模型的平均分为四个部分，并在每个部分的末端进行如图 2 所示的特征映射 logit 蒸馏。最终，CNN 与 Transformer 的特征映射 logit 蒸馏模块得到的蒸馏损失 L_{fpld} 如下所示：

$$L_{fpld} = L_{layer1} + L_{layer2} + L_{layer3} + L_{layer4} \tag{8}$$

在公式(8)中，$L_{layeri} = L_{KD2}$。

2.3. 两阶段蒸馏框架

如图 1(a)所示，使用特征对齐蒸馏模块将 CNN 模型特征层的信息蒸馏到了 Transformer 学生模型中，提高了 Transformer 学生模型提取局部信息的能力。然而，若在整个训练阶段都对 Transformer 学生模型和 CNN 教师模型进行蒸馏，会阻碍 Transformer 学生模型学习自身的归纳偏置[19]。另一方面，根据[7] [22]，由于缺乏一定的归纳偏置，Transformer 网络需要大量的训练数据才能达到与 CNN 网络相当的效果，并且在训练数据充足时，Transformer 网络的效果将优于 CNN 网络。因此，在训练数据充足时，为了充分利用训练数据，尽可能地发挥 Transformer 网络“高容量”的优点，并使其能够学习到自身的归纳偏置，本文提出了一个两阶段蒸馏框架：在第一阶段，使用 CNN 教师模型指导学生模型；在第二阶段，使用 Transformer 教师模型指导学生模型。

如图 1(a) stage1 所示，在第一阶段，使用特征对齐蒸馏模块使 Transformer 学生模型模仿 CNN 教师模型的特征层行为。除了特征层蒸馏，第一阶段也采用 logit 蒸馏：对学生模型的 content token 进行 pool 操作得到 logit 得分，并与教师模型的最后一层 logit 输出进行损失计算。需要注意的是，第一阶段的 logit 蒸馏并没有参照[9]在学生模型中引入额外的蒸馏 token，因为这会使得在下游任务进行微调时引入额外的预训练教师模型。第一阶段的总损失函数如下所示：

$$L_{stage1} = \alpha L_{label} + (1-\alpha) L_{\log it} + \beta L_{fad} \tag{9}$$

在公式(9)中，$L_{label} = D_{CE}(y, p^s)$ 为学生模型的硬标签损失，其中 D_{CE} 为交叉熵损失，y 为真值标签，p^s 为学生模型的预测结果。$L_{\log it} = D_{KL}(p^t, p^s)$ 为 logit 蒸馏损失，其中 D_{KL} 为 KL 散度，p^t 为教师模型的预测结果。L_{fad} 为特征对齐蒸馏模块损失，参照公式(4)。

如图 1(a) stage2 所示，在第二阶段，本文使用一个层数更多的 Transformer 教师模型来指导学生模型。不同的是，在第二阶段，仅进行 logit 蒸馏，同样对学生模型的 content token 进行 pool 操作得到 logit 得分。第一阶段的总损失函数如下所示：

$$L_{stage2} = \alpha L_{label} + (1-\alpha) L_{\log it} \tag{10}$$

需要说明的是，第二阶段的相对位置编码与第一阶段相同。

关于 2.2 节提到的特征映射 log*it* 蒸馏模块，其作为 2.1 节的特征对齐蒸馏模块的替换。因此，若在第一阶段使用特征映射 log*it* 蒸馏模块，则第一阶段的总损失函数如下所示：

$$L_{stage1} = \alpha L_{label} + (1-\alpha) L_{\log it} + \beta L_{fpld} \tag{11}$$

在公式(11)中，L_{fpld} 为特征映射 log*it* 蒸馏模块损失，参照公式(8)。

2.4. 模型实现

按照 2.3 节的设置，本文在训练模型时使用两阶段框架：第一阶段，在 Transformer 学生模型和 CNN 教师模型间进行蒸馏；第二阶段，在 Transformer 学生模型和 Transformer 教师模型间进行蒸馏。

若在第一阶段使用如 3.1 节所表述的特征对齐蒸馏模块，则总体模型如图 1(a)所示，模型记为 TSD-fad。第一阶段，视频帧分别输入 Transformer 学生模型和 CNN 教师模型，并使用特征对齐蒸馏模块对学生模型和教师模型的特征层进行蒸馏。特征对齐蒸馏损失如公式(4)所示，对于教师模型，取最后一个卷积层的输出 $A^t \in R^{h\times w\times c}$ 作为 feature map 大小；对于学生模型，取最后一个编码器的输出 $A^s \in R^{l\times c}$ 作为 content token 大小。需要注意的是，由于使用 2D CNN 模型作为第一阶段的教师模型，在视频帧输入时，需要逐帧输入 Transformer 学生模型进行编码计算。第一阶段的总损失函数为特征对齐蒸馏损失与 log*it* 蒸馏损失之和，如公式(9)所示。第二阶段，视频帧分别输入 Transformer 学生模型和具有相同架构的 Transformer 教师模型。在此阶段，仅进行 log*it* 蒸馏，总损失函数如公式(10)所示。需要注意的是，在第一阶段使用特征对齐蒸馏模块时，需要在 Transformer 学生模型中引入如 2.1 节所述的多头卷积注意力层(*MHCA*)。具体的，原始的 Transformer 学生模型具有 12 个多头自注意层(MHSA)，本文将其中的前 8 个替换为多头卷积注意力层(*MHCA*)。由于分类 token 应该忽略所有其它 token 的位置信息，所以多头卷积注意力层中的相对位置嵌入并不适用于分类 token。因此，参照[19]中的设置，本文使用零向量填充相对位置嵌入并将它们添加到所有 token 中。除此之外，第二阶段的相对位置嵌入与第一阶段保持不变。

若在第一阶段使用如 3.2 节所表述的特征映射 log*it* 蒸馏模块，则第一阶段的模型如图 2 所示，此时总体模型记为 TSD-fpld。特征映射 log*it* 蒸馏损失如公式(8)所示，总损失函数为特征映射 log*it* 蒸馏损失与 log*it* 蒸馏损失之和，如公式(11)所示。

3. 实验

3.1. 实验设置

在本章中，模型在 Something-Something V2 和 Kinetics-400 数据集上进行实验。

预训练策略。本章使用 Kinetics-400 数据集进行预训练，输入视频长度为 16 帧，每帧分辨率大为 224 × 224，使用密集采样[23] [24]，并使用 Multi-ScaleCrop [25]进行数据增强。为了进行两阶段蒸馏，Transformer 学生模型使用 vit-small，它拥有 12 个编码器和 6 个注意力头。在第一阶段，使用经过 pytorch 框架预训练的 resnet101 作为教师模型。在第二阶段，按照[14]中学生模型与视频教师的设置进行知识蒸馏。使用按照[15]设置在 Kinetics-400 上训练 1600 个轮次的 vit-base 作为教师模型，与[14]中计算重建损失不同，在此处仅对学生模型和教师模型的 log*it* 输出计算损失。预训练阶段共 400 个轮次，而对于 2.3 节的两阶段蒸馏框架，消融实验(见下文)表明，第一阶段和第二阶段分别训练 100 个和 300 个轮次时，模型取得最好的效果。在对比实验和消融实验中，除非特别说明，均采用此两阶段蒸馏策略。批大小设置为 1024，使用 Adam W [26]优化器，学习率设置为 1.5e−4，使用余弦衰减[27]，且权重衰减大小为 0.05。

微调和推理策略。微调阶段，将预训练完成的模型应用于下游视频分类任务，包括 Kinetics-400 数据集和 Something-Something V2 数据集。在 Kinetics-400 数据集上微调时，共进行 150 个轮次，使用密集采

样[23] [24]，输入视频为 16 帧，在推理时，使用 3 个空间裁剪和 5 个时间剪辑。在 Something-Something V2 数据集上微调时，共进行 40 个轮次，使用均匀采样[25]，输入视频为 16 帧，在推理时，使用 3 个空间裁剪和 2 个时间剪辑。在两个数据集上均使用 Adam W 优化器[26]，学习率均设置为 1e−3。

Table 1. Comparisons with the other methods on Something-Something V2
表 1. 与其他方法在 Something-Something-V2 数据集上的对比

Model	Pretrain	Crops × Clips	Top-1 (%)	Top-5 (%)
TSM [33]	K400	3 × 2	63.4	88.5
TEINet [35]	IN-1K	1 × 1	62.1	-
TDN [28]	IN-1K	1 × 1	65.3	89.5
ACTION-Net [36]	IN-1K	1 × 1	64.0	89.3
SlowFast R101, 8 × 8 [23]	K400	3 × 1	63.1	87.6
MSNet [37]	IN-1K	1 × 1	64.7	89.4
blvNet [29]	IN-1K	1 × 1	65.2	90.3
Timesformer-HR [34]	IN-1K	3 × 1	62.5	-
ViVit-L/16 × 2 [30]	IN-1K	3 × 1	65.9	89.9
MViT-B, 64 × 3 [38]	K400	3 × 1	67.7	90.9
Mformer-L [39]	K400	3 × 1	68.1	91.2
X-ViT [40]	IN-1K	3 × 1	66.2	90.6
SIFAR-L [41]	K400	3 × 1	64.2	88.4
Video-Swin [31]	K400	3 × 1	69.6	92.7
TPS-T [32]	IN-1K	3 × 1	66.4	90.2
TPS-T [32]	K400	3 × 1	67.9	90.8
TSD-fpld	K400	3 × 2	**70.1**	**92.3**

3.2. 对比实验

表 1 和表 2 分别给出了本文方法在 Something-Something V2 数据集与 Kinetics-400 数据集上与其他现有方法的对比实验结果。需要注意的是，本文提出的方法在 Something-Something V2 数据集上的推理结果，在第一阶段(具体表述见 2.3 节两阶段蒸馏框架)使用特征映射 log*it* 蒸馏模块时(TSD-fpld)达到最佳；而在 Kinetics-400 数据集上的推理结果，在第一阶段(具体表述见 2.3 节两阶段蒸馏框架)使用特征对齐蒸馏模块时(TSD-fad)达到最佳。因此，在表 1 和表 2 中分别只展示 TSD-fpld 和 TSD-fad 的效果，其在相同数据集上的对比结果及分析见 3.3 节。

Table 2. Comparisons with the other methods on Kinetcis-400
表 2. 与其他方法在 Kinetics-400 数据集上的对比

Model	Pretrain	Crops × Clips	Top-1 (%)	Top-5 (%)
I3D [24]	IN-1K	1 × 1	72.1	90.3
NL-I3D [42]	IN-1K	6 × 10	77.7	93.3
CoST [43]	IN-1K	3 × 10	77.5	93.2
SolwFast-R50 [23]	IN-1K	3 × 10	75.6	92.1

续表

X3D-XL [44]	-	3 × 10	79.1	93.9
TSM [33]	IN-1K	3 × 10	74.7	91.4
TEINet [35]	IN-1K	3 × 10	76.2	92.5
TEA [45]	IN-1K	3 × 10	76.1	92.5
TDN [28]	IN-1K	3 × 10	77.5	93.2
Timersformer-L [34]	IN-21K	3 × 1	80.7	94.7
X-ViT [40]	IN-21K	3 × 1	80.2	94.7
MViT-B, 32 × 3 [38]	IN-21K	1 × 5	80.2	94.4
MViT-B, 64 × 3 [38]	IN-21K	3 × 3	81.2	95.1
Mformer-HR [39]	K400	3 × 1	81.1	95.2
TokenShift-HR [46]	IN-21K	3 × 10	80.4	94.5
TPS-T [32]	IN-1K	3 × 4	78.2	92.2
TSD-fad	K400	3 × 5	**81.3**	**94.7**

表 1 的第一栏给出了 TSD-fpld 与 CNN 相关方法的对比结果。slowfast [23]利用快慢双通道分别处理视频帧中的时间信息和空间信息。TDN [28]通过短时建模和长时建模分别融合段内局部运动变化信息和增强段间运动变化信息。blVNet [29]构造了浅层网络和深层网络来分别处理高分辨率帧和低分辨率帧，并使用一个深度卷积来提取视频帧中的时序信息。相比于这些方法使用不同模块分别处理视频帧的空间信息和时间信息，TSD-fpld 通过设置两阶段蒸馏框架直接学习不同教师模型的预测分布：第一阶段，让能有效地提取图像中局部特征的 CNN 教师指导学生模型；第二阶段，让能有效地建模长距离依赖关系的 Transformer 教师指导学生。通过直接模仿教师模型的预测分布，学生模型可以减少参数优化的时间，从而加快训练速度。

表 1 的第二栏给出了 TSD-fpld 与 Transformer 相关方法的对比结果。ViVit [30]研究了将原始 vit 引入视频任务所得到的几种变体，如将编码器拆解为空间编码器和时间编码器来分别处理空间和时间信息。Video-Swin [31]通过将自注意力计算限制在窗口内来减小计算量，并提出偏移窗口注意力计算进行全局特征融合。TPS [32]通过在不同视频帧之间进行 patch 块移位，使得当前视频帧包含其他帧的部分内容，增大了当前帧的时间感受野。而本文提出的 TSD-fpld 通过特征映射 log*it* 蒸馏模块将学生模型的特征映射为 log*it*，并与 CNN 教师模型的最终 log*it* 输出计算损失，使学生模型的特征层学习到了教师模型的预测分布，同时也实现 CNN 与 Transformer 之间的通用知识蒸馏。即便在与时序信息高度相关的 Something-Something V2 数据集上，TSD-fpld 的效果也优于上述方法。

在表 2 中，TSD-fad 首先与一些使用 3D CNN 或分解(2 + 1) D CNN 的方法进行对比，如表 2 第一栏所示。TSM [33]是一种是用卷积进行视频分类的高效方法，其通过在时间维度上移动部分通道，实现相邻帧之间的信息交换。相比于 TSM，TSD-fad 使用原始的 vit-small，并通过将其中的前 8 个多头自注意力层(MHSA)替换为包含相对位置信息的多头卷积注意力层(*MHCA*)，使得 TSD-fad 在学习非局部语义依赖的同时能够意识到局部细节。并且，由于 Transformer 模型的“容量”高于 CNN 模型，可以看到，在经过 400 个轮次的预训练和 40 个轮次的微调后，TSD-fad 的预测准确率高于 TSM 及其他 CNN 方法。

表 2 的第二栏给出了 TSD-fad 与 Transformer 相关方法的对比结果。Timesformer-L [34]抛弃了 CNN，使用纯 Transformer 架构，并探讨了包含联合时空自注意力在内的几种注意力计算方式。但是，Timesformer-L 的纯 Transformer 结构使其难以有效提取视频帧中的局部特征。因为相比于 Transformer，

CNN 由于卷积运算的局部性和权值共享机制，具有很强的归纳偏置，能够有效地提取图像中的局部特征[19]。相比于 Timesformer 的纯 Transformer 架构，TSD-fad 通过在第一阶段引入 CNN 教师指导学生模型，迫使学生模型模仿 CNN 教师模型特征层的操作，提高了学生模型局部建模的能力。因此在与场景高度相关的 Kinteics-400 数据集上，TSD-fad 的效果优于 Timesformer。

3.3. 消融实验

在本节中，通过设置多组消融实验以验证各个模块的有效性，消融实验结果见表 3~6。其中，表 3 所述实验在 Something-Something V2 和 Kinetics-400 数据集上进行，表 4 和表 5 所述实验仅在 Something-Something V2 数据集上进行，表 6 所述实验仅在 Kinetics-400 数据集上进行。

(1) 本文研究了不同的特征蒸馏模块对模型的影响。结果如表 3 所示，其中 TSD-fad 和 TSD-fpld 分别表示在第一阶段使用特征对齐蒸馏模块和特征映射 log*it* 蒸馏模块，(w/o)后缀表示不使用对应模块。比较表 3 的第一、第二行所示结果可以看到：在第一阶段使用特征对齐蒸馏模块使本文所述方法分别在 Something-Something V2 和 Kinetics-400 数据集上取得了 0.6%和 0.3%的准确率提升。而比较表 3 的第三、第四行所示结果可以看到：在第一阶段使用特征对齐蒸馏模块使本文所述方法分别在 Something-Something V2 和 Kinetics-400 数据集上取得了 0.5%和 0.4%的准确率提升。验证了特征对齐蒸馏模块和特征映射 log*it* 蒸馏模块的有效性。而比较表 3 的第二、第四行可以得到：在 Something-Something V2 数据集上使用特征映射 logit 蒸馏模块的效果优于特征对齐蒸馏模块，在 Kinetics-400 数据集上使用特征对齐蒸馏模块的效果优于特征映射 logit 蒸馏模块。两个特征蒸馏模块表现出了明显的数据集偏向性：Kinetics-400 数据集与场景特征高度相关，相邻视频帧在时间维度上变化不大，而特征对齐蒸馏模块直接使学生模型的特征层模仿 CNN 教师模型的特征层行为，因而更好地提取到了局部特征，取得了更高的准确率；而 Something-Something V2 数据集对时序信息尤其敏感，相比于关注局部特征的特征对齐蒸馏模块，特征映射 log*it* 蒸馏模块直接使学生模型学习教师模型的预测分布，因而取得了更高的准确率。

(2) 表 4 给出了特征映射 log*it* 蒸馏模块(TSD-fpld)的特征蒸馏层数对模型预测效果的影响。在 2.2 节中，本文方法将学生模型的编码器平均分为四个部分，并在每部分的末层构造特征映射 log*it* 分支与教师

Table 3. The effect of feature distillation module on Something-Something V2 and Kinetics-400
表 3. 特征蒸馏模块在 Something-Something V2 和 Kinetics-400 数据集上的影响

	Something-Something V2	Kinetics-400
TSD-fpld (w/o)	69.5	80.7
TSD-fpld	70.1	81.0
TSD-fad (w/o)	69.1	80.9
TSD-fad	69.6	81.3

Table 4. The effect of TSD-fpld's feature distillation layers on Something-Something V2
表 4. TSD-fpld 的特征蒸馏层数在 Something-Something V2 数据集上的影响

Layer4	Layer3	Layer2	Layer1	Top-1 (%)
				69.5
√				69.9
√	√			70.0
√	√	√		70.0
√	√	√	√	**70.1**

Table 5. The effect of TSD-fpld's stage1 epochs on Something-Something V2
表 5. TSD-fpld 的第一阶段训练轮次在 Something-Something V2 数据集上的影响

Stage1 epochs	0	50	75	100	125	150
Top-1 (%)	69.3	69.8	69.9	**70.1**	69.7	69.5

Table 6. The effect of the loss function in TSD-fad on Kinetics-400
表 6. TSD-fad 的损失函数在 Kinetics-400 数据集上的影响

Lfeature	Llabel (stage1)	Llog*it* (stage1)	Llabe l (stage2)	Llog*it* (stage2)	Top-1 (%)
			√	√	80.5
		√	√	√	80.8
	√	√	√	√	80.9
√	√	√	√	√	**81.3**

模型进行蒸馏，四个分支分别记为 Layer1、Layer2、Layer3 和 Layer4，如表 4 所示。当增加 Layer4 蒸馏分支时，模型取得了 0.4%的准确率提升；而当增加 Layer1、Layer2 和 Layer3 蒸馏分支时，模型的准确率则变化不大。特征映射 log*it* 蒸馏模块直接使学生模型学习教师模型的预测分布，而 Transformer 的深层编码器(Layer4)由于经过多层的堆叠，能够将低级特征与更高级的上下文信息结合，产生更有表现力的特征表示，因此在使用 Layer4 蒸馏分支时，模型能够取得较大的准确率提升。尽管如此，在使用特征映射 log*it* 蒸馏模块时，本文方法仍然使用四个蒸馏分支。

(3) 表 5 给出了在第一阶段使用特征映射 log*it* 蒸馏模块(TSD-fpld)时训练轮次对模型预测效果的影响。需要注意的是，第一阶段与第二阶段的总训练轮次始终为 400。当 Stage1 epochs 为 0 时，表示抛弃了两阶段蒸馏框架的第一阶段，只在第二阶段对模型预训练 400 个轮次，损失函数见公式(10)。此时，模型取得了 69.3%的准确率，而当使用两阶段蒸馏框架且在第一阶段训练 100 个轮次时，模型的准确率达到了 70.1%，取得了 0.8%的准确率提升，验证了两阶段蒸馏框架的有效性：第一阶段，特征蒸馏使学生模型学习到了 CNN 教师的局部特征建模能力；第二阶段，简单的 log*it* 蒸馏使学生模型学习到了 Transformer 教师的长距离依赖关系建模能力。而对于其他的训练轮次设置：较少的第一阶段训练轮次无法使 Transformer 学生模型学习到足够的局部建模能力；而较多的第一阶段训练轮次则阻碍了 Transformer 学生模型学习自身的归纳偏置(在训练数据充足时，Transformer 的效果优于 CNN)。

(4) 表 6 给出了在第一阶段使用特征对齐蒸馏模块(TSD-fad)时损失函数对模型预测效果的影响。表 6 中各损失函数参见 2.3 节。对比表 6 的第一、第二行可以看到：在使用两阶段蒸馏框架时(第一阶段仅对学生模型和教师模型的 log*it* 输出进行蒸馏)，模型取得了 0.3%的准确率提升。而对比第二、第三行则验证了第一阶段硬标签蒸馏的有效性，第三、第四行对比验证了特征对齐蒸馏模块的有效性。

4. 结论

本文针对 Transformer 网络缺少一定的归纳偏置的问题，提出了特征对齐蒸馏模块。该模块通过将 Transformer 的特征层与 CNN 的特征层进行维度对齐，实现了 Transformer 与 CNN 的特征层蒸馏，使 Transformer 学习到了 CNN 的局部建模能力。针对特征对齐操作会引入卷积操作增加模型计算量的问题，本文又提出了特征映射 log*it* 蒸馏模块，通过将 Transformer 的特征层映射为 log*it*，实现了 Transformer 与 CNN 特征层的通用蒸馏方法。为了使学生模型同时学习局部建模能力和长距离依赖建模能力，本文提出了两阶段蒸馏框架，实现了 CNN 教师和 Transformer 教师对学生模型的协同指导。本文提出的方法在 Something-Something V2 和 Kinetics-400 数据集上分别取得了 70.1%和 81.3%的 Top-1 准确率，具有良好

的性能。

基金项目

安徽省自然科学基金项目(2408085MF157)。

参考文献

[1] Gordon, D.F. and Desjardins, M. (1995) Evaluation and Selection of Biases in Machine Learning. *Machine Learning*, **20**, 5-22. https://doi.org/10.1007/bf00993472

[2] Goyal, A. and Bengio, Y. (2020) Inductive Biases for Deep Learning of Higher-Level Cognition. arXiv: 2011.15091.

[3] Vaswani, A., Shazeer, N., Parmar, N., *et al*. (2017) Attention Is All You Need. arXiv: 1706.03762.

[4] Wang, W., Xie, E., Li, X., Fan, D., Song, K., Liang, D., *et al*. (2021) Pyramid Vision Transformer: A Versatile Backbone for Dense Prediction without Convolutions. 2021 *IEEE*/*CVF International Conference on Computer Vision* (*ICCV*), Montreal, 10-17 October 2021, 548-558. https://doi.org/10.1109/iccv48922.2021.00061

[5] Dai, Z.H., Liu, H.X., Le, Q.V. and Tan, M.X. (2021) CoAtNet: Marrying Convolution and Attention for All Data Sizes. arXiv: 2106.04803.

[6] Beal, J., Kim, E., Tzeng, E., Park, D.H., Zhai, A. and Kislyuk, D. (2020) Toward Transformer-Based Object Detection. arXiv: 2012.09958.

[7] Dosovitskiy, A., Beyer, L., Kolesnikov, A., *et. al*. (2021) An Image Is Worth 16 × 16 Words: Transformers for Image Recognition at Scale. arXiv: 2010.11929.

[8] Ren, S., He, K., Girshick, R. and Sun, J. (2017) Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **39**, 1137-1149. https://doi.org/10.1109/tpami.2016.2577031

[9] Touvron, H., Cord, M., Douze, M., *et al*. (2021) Training Data-Efficient Image Transformers & Distillation through Attention. *Proceedings of the* 38*th International Conference on Machine Learning*, 18-24 July 2021, 10347-10357.

[10] Xiao, T., Singh, M., Mintun, E., Darrell, T., Dollár, P. and Girshick, R.B. (2021) Early Convolutions Help Transformers See Better. arXiv: 2106.14881.

[11] Hao, Z.W., Guo, J.Y., Han, K., *et al*. (2023) One-for-All: Bridge the Gap between Heterogeneous Architectures in Knowledge Distillation. arXiv: 2310.19444.

[12] Cortes, C., Mohri, M. and Rostamizadeh, A. (2012) Algorithms for Learning Kernels Based on Centered Alignment. *Journal of Machine Learning Research*, **13**, 795-828.

[13] Kornblith, S., Norouzi, M., Lee, H. and Hinton, G.E. (2019) Similarity of Neural Network Representations Revisited. *Proceedings of the* 36*th International Conference on Machine Learning*, Long Beach, 9-15 June 2019, 3519-3529.

[14] Wang, R., Chen, D., Wu, Z., Chen, Y., Dai, X., Liu, M., *et al*. (2023) Masked Video Distillation: Rethinking Masked Feature Modeling for Self-Supervised Video Representation Learning. 2023 *IEEE*/*CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), Vancouver, 17-24 June 2023, 6312-6322. https://doi.org/10.1109/cvpr52729.2023.00611

[15] Tong, Z., Song, Y.B., Wang, J. and Wang, L.M. (2022) VideoMAE: Masked Autoencoders Are Data-Efficient Learners for Self-Supervised Video Pre-Training. arXiv: 2203.12602.

[16] He, K.M., Chen, X.L., Xie, S.N., Li, Y.H., Dollár, P. and Girshick, R.B. (2022) Masked Autoencoders Are Scalable Vision Learners. *IEEE*/*CVF Conference on Computer Vision and Pattern Recognition*, New Orleans, 19-20 June 2022, 15979-15988.

[17] Cordonnier, J.B., Loukas, A. and Jaggi, M. (2020) On the Relationship between Self-Attention and Convolutional Layers. arXiv: 1911.03584.

[18] Parmar, N., Ramachandran, P., Vaswani, A., *et al*. (2019) Stand-Alone Self-Attention in Vision Models. *Advances in Neural Information Processing Systems*, **32**, 68-80.

[19] Chen, X., Cao, Q., Zhong, Y., Zhang, J., Gao, S. and Tao, D. (2022) DearKD: Data-Efficient Early Knowledge Distillation for Vision Transformers. 2022 *IEEE*/*CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), New Orleans, 18-24 June 2022, 12042-12052. https://doi.org/10.1109/cvpr52688.2022.01174

[20] Romero, A., Ballas, N., Ebrahimi Kahou, S., *et al*. (2015) FitNets: Hints for Thin Deep Nets. arXiv: 1412.6550.

[21] Hinton, G.E., Vinyals, O. and Dean, J. (2015) Distilling the Knowledge in a Neural Network. arXiv: 1503.02531.

[22] d'Ascoli, S., Touvron, H., Leavitt, M.L., *et al*. (2021) Convit: Improving Vision Transformers with Soft Convolutional Inductive Biases. *Proceedings of the* 38*th International Conference on Machine Learning*, 18-24 July 2021, 2286-2296.

[23] Feichtenhofer, C., Fan, H., Malik, J. and He, K. (2019) SlowFast Networks for Video Recognition. 2019 *IEEE*/*CVF International Conference on Computer Vision* (*ICCV*), Seoul, 27 October-2 November 2019, 6201-6210. https://doi.org/10.1109/iccv.2019.00630

[24] Carreira, J. and Zisserman, A. (2017) Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. 2017 *IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), Honolulu, 21-26 July 2017, 4724-4733. https://doi.org/10.1109/cvpr.2017.502

[25] Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., *et al*. (2019) Temporal Segment Networks for Action Recognition in Videos. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **41**, 2740-2755. https://doi.org/10.1109/tpami.2018.2868668

[26] Loshchilov, I. and Hutter, F. (2017) Fixing Weight Decay Regularization in Adam. arxiv: 1711.05101.

[27] Loshchilov, I. and Hutter, F. (2017) SGDR: Stochastic Gradient Descent with Warm Restarts. arXiv: 1608.03983.

[28] Wang, L., Tong, Z., Ji, B. and Wu, G. (202). TDN: Temporal Difference Networks for Efficient Action Recognition. 2021 *IEEE*/*CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), Nashville, 20-25 June 2021, 1895-1904. https://doi.org/10.1109/cvpr46437.2021.00193

[29] Fan, Q.F., Chen, C.F., Kuehne, H., *et al*. (2019) More is Less: Learning Efficient Video Representations by Big-Little Network and Depthwise Temporal Aggregation. arXiv: 1912.0086.

[30] Arnab, A., Dehghani, M., Heigold, G., Sun, C., Lucic, M. and Schmid, C. (2021) ViViT: A Video Vision Transformer. 2021 *IEEE*/*CVF International Conference on Computer Vision* (*ICCV*), Montreal, 10-17 October 2021, 6816-6826. https://doi.org/10.1109/iccv48922.2021.00676

[31] Liu, Z., Ning, J., Cao, Y., Wei, Y., Zhang, Z., Lin, S., *et al*. (2022) Video Swin Transformer. 2022 *IEEE*/*CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), New Orleans, 18-24 June 2022, 3192-3201. https://doi.org/10.1109/cvpr52688.2022.00320

[32] Xiang, W., Li, C., Wang, B., Wei, X., Hua, X. and Zhang, L. (2022) Spatiotemporal Self-Attention Modeling with Temporal Patch Shift for Action Recognition. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M. and Hassner, T., Eds., *Computer Vision—ECCV* 2022., Springer, 627-644. https://doi.org/10.1007/978-3-031-20062-5_36

[33] Lin, J., Gan, C. and Han, S. (2019) TSM: Temporal Shift Module for Efficient Video Understanding. 2019 *IEEE*/*CVF International Conference on Computer Vision* (*ICCV*), Seoul, 27 October-2 November 2019, 7082-7092. https://doi.org/10.1109/iccv.2019.00718

[34] Bertasius, G., Wang, H. and Torresani, L. (2021) Is Space-Time Attention All You Need for Video Understanding? arXiv: 2102.05095.

[35] Liu, Z., Luo, D., Wang, Y., Wang, L., Tai, Y., Wang, C., *et al*. (2020) TEINet: Towards an Efficient Architecture for Video Recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, **34**, 11669-11676. https://doi.org/10.1609/aaai.v34i07.6836

[36] Wang, Z., She, Q. and Smolic, A. (2021) Action-Net: Multipath Excitation for Action Recognition. 2021 *IEEE*/*CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), Nashville, 20-25 June 2021, 13209-13218. https://doi.org/10.1109/cvpr46437.2021.01301

[37] Kwon, H., Kim, M., Kwak, S. and Cho, M. (2020) MotionSqueeze: Neural Motion Feature Learning for Video Understanding. In: Vedaldi, A., Bischof, H., Brox, T. and Frahm, J.M., Eds., *Computer Vision—ECCV* 2020, Springer, 345-362. https://doi.org/10.1007/978-3-030-58517-4_21

[38] Fan, H., Xiong, B., Mangalam, K., Li, Y., Yan, Z., Malik, J., *et al*. (2021) Multiscale Vision Transformers. 2021 *IEEE*/*CVF International Conference on Computer Vision* (*ICCV*), Montreal, 10-17 October 2021, 6804-6815. https://doi.org/10.1109/iccv48922.2021.00675

[39] Patrick, M., Campbell, D., Asano, Y.M., *et al*. (2021) Keeping Your Eye on the Ball: Trajectory Attention in Video Transformers. *Advances in Neural Information Processing Systems*, **34**, 12493-12506.

[40] Bulat, A., Pérez-Rúa, J.M., Sudhakaran, S., *et al*. (2021) Space-Time Mixing Attention for Video Transformer. *Advances in Neural Information Processing Systems*, **34**, 19594-19607.

[41] Fan, Q.F., Chen, C.F. and Panda, R. (2021) An Image Classifier Can Suffice for Video Understanding. arXiv: abs/2106.14104.

[42] Wang, X.L., Girshick, R.B., Gupta, A. and He, K.M. (2017) Non-Local Neural Networks. arXiv: 1711.07971.

[43] Li, C., Zhong, Q.Y., Xie, D. and Pu, S.L. (2019) Collaborative Spatiotemporal Feature Learning for Video Action Recognition. *IEEE Conference on Computer Vision and Pattern Recognition* 2019, Long Beach, 15-20 June 2019, 7872-7881.

[44] Feichtenhofer, C. (2020) X3D: Expanding Architectures for Efficient Video Recognition. 2020 *IEEE*/*CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), Seattle, 13-19 June 2020, 200-210. https://doi.org/10.1109/cvpr42600.2020.00028

[45] Li, Y., Ji, B., Shi, X., Zhang, J., Kang, B. and Wang, L. (2020) TEA: Temporal Excitation and Aggregation for Action Recognition. 2020 *IEEE*/*CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), Seattle, 13-19 June 2020, 906-915. https://doi.org/10.1109/cvpr42600.2020.00099

[46] Zhang, H., Hao, Y. and Ngo, C. (2021) Token Shift Transformer for Video Classification. *Proceedings of the* 29*th ACM International Conference on Multimedia*, 20-24 October 2021, 917-925. https://doi.org/10.1145/3474085.3475272